

AD-A060 736

NAVAL RESEARCH LAB WASHINGTON D C
PATTERN RECOGNITION WITH PARTLY MISSING DATA.(U)
AUG 78 J K DIXON

F/6 9/4

UNCLASSIFIED

NRL-MR-3825

SBIE-AD-E000 230

NL

1 OF 1

AD
A060736



END
DATE
FILMED
01-79

DDC

DDC FILE COPY AD A060736

(12)

NW

Ad 000 230

NRL Memorandum Report 3825

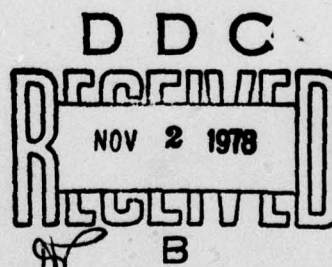
Pattern Recognition with Partly Missing Data

John K. Dixon

*Computer Science Laboratory
Communications Sciences Division*

LEVEL II

August 1978



78 10 05 028

NAVAL RESEARCH LABORATORY
Washington, D.C.

Approved for public release; distribution unlimited.

14

NRL-MR-3825

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Memorandum Report 3825	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) PATTERN RECOGNITION WITH PARTLY MISSING DATA		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL Problem.
7. AUTHOR(s) John K. Dixon		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, D. C. 20375		8. CONTRACT OR GRANT NUMBER(s) Memorandum rept.
11. CONTROLLING OFFICE NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NRL Problem B02-23 61153N-14, RR014-02, RR014-02-41
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 2pp.		12. REPORT DATE Aug 1978
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		13. NUMBER OF PAGES 19
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
18. SUPPLEMENTARY NOTES		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Pattern Recognition Blanks in Data Nearest Neighbor Blanks		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This paper is an experimental comparison of several simple, cheap ways of doing pattern recognition when some data elements are missing (blank). Pattern recognition methods are usually designed to deal with perfect data, but in the real world data elements are often missing due to error, equipment failure, change of plans, etc. Six methods of dealing with blanks are tested on five data sets. Blanks were inserted at random locations into the data sets. A version of the K-nearest-neighbor technique was used to classify the (Continues)		

DD FORM 1473 1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF 014-6601

1

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251 950

78 10 05 028

next page
Jhu

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

data and evaluate the six methods. Two methods were found to be consistently poor. Four methods were found to be generally good. Suggestions are given for choosing the best method for a particular application.

CONTENTS

1. INTRODUCTION	1
2. HUMAN INTUITION	3
3. SIX METHODS OF HANDLING BLANKS	4
4. DATA SETS	7
5. EXPERIMENTAL METHOD	8
6. EXPERIMENTAL RESULTS	9
7. CONCLUSIONS	11
REFERENCES	13 & 14
TABLES	15 & 16

ACCESSION for		
NTIS	White Section	<input checked="" type="checkbox"/>
DDC	Buff Section	<input type="checkbox"/>
UNANNOUNCED		<input type="checkbox"/>
JUSTIFICATION _____		
BY _____		
EXTENSION/AVAILABILITY CODES		
Dist. AVAIL. and/or SPECIAL		
A		

PATTERN RECOGNITION WITH PARTLY MISSING DATA

I. INTRODUCTION

When computer pattern recognition methods are tested in the Laboratory, care is usually taken to see that the experimental data sets used are complete and free from error. However, when pattern recognition is applied to practical problems we often find that real world data contains many missing values. When a human pattern recognizer is faced with imperfect data he does the best he can using the data that is available. This kind of human flexibility is so natural that we expect it and think little of it.

A program which demands perfect data is not well matched to human information systems where some degree of error is acceptable. But a program which can accept imperfect data, while still performing reasonably well, fits in much better and is more economically useful.

Data may contain missing values for a number of reasons. A person filling out a questionnaire may neglect or refuse to answer certain questions. An intercepted radar pulse may be so weak and noisy that some of the more subtle features are not measurable. The method of collecting data may change during the course of an investigation so that a feature is measured in some cases but not in others. Several similar but not identical data sets from different sources may be combined. Malfunctioning equipment may collect some features of an event but not others. A human operator may collect detailed information when he has time, but only the most essential features when he is busy.

The problem then is this:

We have a set of M objects. We wish to identify or classify these objects by means of a series of N measurements performed on each object. Each set of N measurements forms a vector. The complete set of vectors forms an $M \times N$ matrix. But some of the measurement numbers may be missing. They are replaced by symbols which indicate blanks. We wish to classify the vectors

containing blanks in such a way that we get the same classification results, as nearly as possible, as would be obtained with the complete data.

The experimental design is to take several sets of data and apply a form of nearest neighbor pattern recognition [2, 4, 15]. Then blanks are inserted at random into the data, and pattern recognition is done again by several different experimental methods. By comparing the recognition scores, we can see how well the various methods work.

The methods for handling blanks used in this paper are applicable to many other techniques of pattern recognition and clustering. Some methods first correct the data by filling in blanks with estimated values. These methods could be used as a first step with any form of pattern recognition or clustering. Other methods involve estimating the distance between two vectors which contain blanks. These methods could be used with any form of pattern recognition or clustering which is based on computing the distance between two vectors.

In this paper we shall concentrate on methods which are simple, cheap, easy to implement, and do not depend on special properties of the data.

By using special properties of the data, we would expect to get better results in some cases. For example, it might be that some missing value is a function of other values. Thus, it would be possible to recompute the missing value exactly. We do not consider such methods here because they depend on the particular properties of the data base and may thus be of less general interest. Moreover, a simple method may be more economical if its performance is adequate.

The problem we attack here is not the same as that of finding errors in data, a problem which Lee [9] and Hammer [6] have considered. The problem of missing data has been previously considered by Lee, Slagle, and Mong [9]. In fact a modification of Lee's method is one of those used in this study. However, the experimental procedure of Lee is different. He used only one method (a good method by our results) to estimate missing data and then compared the estimated value with the original value to see what percentage of error there was in the estimate.

One should also note the work of Skinner [12] on a generalized method of answering questions and estimating missing values in a data set. The philosophy of Skinner's method is similar to that of Lee, et al. The major difference is that Skinner deals with non-numerical data.

Sebestyen [11] (p.74) gives a theoretical analysis of pattern classification with partly missing data. Sebestyen assumes that the conditional probability of class membership given any set or partial set of measurements is known. Whereas, we assume only that the probabilities of class membership are defined by a collection of data which contains many missing values. In general, one might say that Sebestyen gives a theoretical analysis of the missing data problem based on some idealistic assumptions. We, on the other hand, give an experimental study based on more realistic assumptions.

Nevertheless, Sebestyen's theoretical analysis is of some interest. Because of the different assumptions, the decision rules considered by Sebestyen do not correspond in any clear way to the methods used here. The final conclusion of Sebestyen is that one should use only the measured values; no useful purpose is served by attempting to estimate the missing values. This conclusion can be applied to the present study only by analogy.

II. HUMAN INTUITION

Humans are still the best problem solvers in a great many areas, therefore it makes sense to apply human intuition to the blank data problem to gain insight. Let us perform the following gedanken experiment. Consider the following two 5-dimensional vectors, A and B:

A = (1.2 3.7 10.9 6.3 5.9)
B = (1.1 3.5 Blank 6.2 5.7)

What does intuition suggest about the blank value? One tends to feel that the blank value is close to 10.9 since all the other values are close. More precisely, one might estimate that the blank is a little smaller than 10.9 since all the other B components are a little smaller than the corresponding A components. Of course, our intuition would also depend on other vectors in the data set which are not shown here.

There are several ways to interpret this intuitive feeling:

1. An assumption that the data are clustered.
2. An assumption that the various features are correlated so that one can do linear interpolation across rows (columns).
3. An assumption that if distances are small along each of 4 dimensions, they will also be small along the 5th.

III. SIX METHODS OF HANDLING BLANKS

There are many methods of handling blanks that seem intuitively reasonable. In this study we will compare six methods. These methods either eliminate blanks by deleting part of the data, or fill in the blanks with estimated values, or compute an estimated distance between vectors which contain blanks.

We will use the following notation:

V_i is a vector with N components or features, $X_{i,j}$

$V_i = (X_{i,1}, X_{i,2}, X_{i,3}, \dots, X_{i,N})$

DAB_j is the component of distance between vectors A & B

along the j th feature:

$DAB_j = X_{a,j} - X_{b,j}$

We assume that the entire data set includes M vectors, and that it has been normalized so that each feature or column of the data matrix has zero mean and unit standard deviation.

The six methods used in this study are described below:

1. LEE⁴

This is a modification of the method suggested by Lee, Slagle, and Mong [9]. If a vector A has a blank in the j th feature find the four nearest neighbors of A , then average the j th feature of the four nearest neighbors, and fill in the blank with this value. Lee assumes that blanks occur only in the one vector being filled in. If we assume that the data base contains many blanks, then we must have a way of computing distance between vectors containing arbitrary blanks. In this work we have used the method described for NORMAL below. Also, this program follows the rule that if a vector has a blank in the j th position, it is discarded. Thus the four nearest neighbors found will not have blanks in the j th feature. Since LEE⁴ and NORMAL use the same method of finding nearest neighbors, they are closely related. However, because things are done in a different order, they do not, in general, get the same answers. Also Lee requires that the four nearest neighbors be members of the same cluster. We do not have such a

requirement, although in most cases they probably will be the same vectors as selected by the Lee method.

2. LEE1

This is the same as the LEE4 method described above except that only one nearest neighbor is used instead of four. White [15] finds that a small K gives better results when the data base is small. K is the number of nearest neighbors used.

3. DELETE

This program simply deletes vectors or features which contain blanks. In order to minimize the waste of good data, the following heuristic procedure is used: The percentage of blanks in every row and every column is computed. Then the row or column with the highest percentage of blanks is deleted. Next, the percentages are recomputed and another row or column is deleted. The process is repeated until all blanks are eliminated. This method has the advantage that it produces perfect data without making any assumptions. The obvious disadvantage is that a lot of good data is wasted. Intuitively, this seems a very poor method, yet in the real world of practical computer programs, some form of deletion is often used to deal with blanks; see Andrews [1].

4. NORMAL

This program computes the distance between two vectors and then normalizes to compensate for blanks. For example, suppose we have two vectors A and B. Suppose NB is the number of features which are blank (in one vector or the other or both), then the distance between vectors is computed as follows:

$$DAB_j = \begin{cases} 0 & \text{if } X_{a,j} \text{ or } X_{b,j} \text{ is blank} \\ (X_{a,j} - X_{b,j})^2 & \text{otherwise} \end{cases}$$

$$DISTANCE_{ab} = \frac{N}{N-NB} \sum_{j=1}^N (DAB_j)^2$$

The result is the square of Euclidean distance if there are no blanks.

Another way to express the action of this program is that it assumes the distance to a blank feature is the same as the average distance between non-blank features of the same pair of vectors. This method seems intuitively reasonable in light of the gedanken experiment in Section II.

5. AVERAGE

This program assumes that the distance to a blank is the same as the average distance between all pairs of vectors along that one feature:

$$A_j^* = \frac{2}{M(M-1)} \sum_{i=2}^M \sum_{k=1}^{i-1} |X_{i,j} - X_{k,j}|$$

All the A^* are computed and saved in a table. Then as computation proceeds, whenever it is necessary to compute the distance to a blank, the A^* value is selected from the table. In other words, the distance from V_a to V_b is:

$$DAB_j = \begin{cases} A_j^* & \text{if } X_{a,j} \text{ or } X_{b,j} \text{ is blank} \\ (X_{a,j} - X_{b,j}) & \text{otherwise} \end{cases}$$

$$DISTANCE_{ab} = \left[\sum_{j=1}^N (DAB_j)^2 \right]$$

6. ZERO

This program assumes that the distance to any blank is zero:

$$DAB_j = \begin{cases} 0 & \text{if } X_{a,j} \text{ or } X_{b,j} \text{ is blank} \\ (X_{a,j} - X_{b,j}) & \text{otherwise} \end{cases}$$

$$DISTANCE_{ab} = \sum_{j=1}^N (DAB_j)^2$$

This is the same as the NORMAL program except that normalization is not done. This means that a vector with a great many blanks will seem to be very close to other vectors.

IV. DATA SETS

While a theoretical analysis of these various methods might be useful, it would be very difficult to decide what assumptions to make about the statistical properties of the data. Therefore, we have chosen an experimental approach.

Five data sets were selected. All of these data sets are free of blanks except one, which contains a small number. Blanks were inserted into the data sets at locations selected by a random number generator.

1. IRIS

This data consists of the first 20 vectors of each class from the famous iris data first used by Fisher [5]. $M = 60$, $N = 4$ there are 3 classes, of 20 vectors each.

2. B22

This data consists of features measured on some radar signals. $M = 335$, $N = 43$. There are 14 classes. This data set contains 2.78% "natural" blanks.

3. VOICE

This data consists of human speech in the form of linear prediction coefficients. Each vector is a set of 10 coefficients. These vectors were generated from human speech at a sampling rate of 44 vectors per second. Each vector has been assigned to a phoneme class (or silence) by human inspection of the LPC output along with knowledge of what was said. $M = 100$, $N = 10$ and there are 14 classes.

4. CAR

This is an artificial data set constructed by evaluation of an algebraic expression in six variables:

$$Y = X_1^2 - 2X_1X_2X_3 + X_2^2X_3 + X_3^2 - 2X_4X_5X_6 + X_5^2X_6$$

Arguments for the expression were constructed by a random

number generator. Then the algebraic expression was evaluated for each vector and each vector was assigned to a class depending on the value of Y . The range of Y was partitioned into 5 intervals. The boundaries of the 5 intervals were selected by hand so as to make 5 classes of approximately equal size. $M = 150$, $N = 6$.

5. WPS

This data set is taken from the forms used by judges to evaluate scientific papers submitted to the World Population Society (an interdisciplinary scientific society). The judges are asked to evaluate each paper on a scale of one to five in five different areas. The areas are: new results, scientific value, relevance to the population field, interdisciplinary character, and an overall rating. Overall rating was used as the class, thus there are 5 classes. $M = 145$, $N = 5$. The serial number of each paper is also part of the data in order to see if earliness or lateness had any effect on the quality of the paper.

V. EXPERIMENTAL METHOD

Various numbers of blanks (usually 20% or 30%) were inserted into the data sets. Then a jackknife test was performed on each data set by each of the six programs.

The jackknife test is a way of dividing the data set into a learning set and an experimental set. One vector V is selected from the data set. The program pretends that the class of V is unknown and attempts to classify it using the remaining $M-1$ vectors. This procedure is repeated until every vector in the data set has played the role of unknown. Since the correct classes are known, the program can score itself by computing the percentage of correct classifications.

In the case of the DELETE program, the vectors which are deleted are not counted in scoring. For example if the data set has 100 vectors, and 40 are deleted, and 30 of the remaining 60 are correctly classified, the score is 50%.

The classification procedure is a distance-weighted K-nearest neighbor method with $K = 4$. If the K-nearest neighbors of some vector V , are all of the same class, C , then V is assumed to belong to class C . However, if the neighbors are of different classes then a voting procedure is used. Each neighbor votes for its class with a weight inversely proportional to its distance from V .

The decision rule used here is very similar to the one described by Dudani [4]. The only difference is that a different

weighting function is used. Dudani showed that the distance-weighted K-nearest neighbor decision rule is superior to majority rating.

VI. EXPERIMENTAL RESULTS

Some experimental results are given in Table 1. These are recognition scores expressed as percentages. The first column, labeled "pure" is the score obtained when there are no blanks in the data (except the B22 data set, which has 2.8% "natural" blanks). The second column, labeled "chance", is the expected score one would obtain if a random choice of the nearest neighbor ($k=1$) was made for each attempt at classification. This score is computed from the number of classes and the number of vectors in each class, by the formula given below.

$$P = \sum_{i=1}^K \frac{N_i(N_i-1)}{M(M-1)}$$

Where:

- P = probability of correct classification by chance
- K = number of distinct classes
- N = number of elements in class i
- M = number of elements in the entire data set

This is the score one would expect if the data were 100% blank. The other six columns in table 1 correspond to the six different programs.

Each row corresponds to a different data set. Each data set has been used four times with different blanks inserted. In some cases the percentage of blanks was changed; in some cases the locations of the blanks were changed by changing the parameters of the random number generator. Of course the percentage of blanks information does not apply to the PURE column or the CHANCE column.

As one can see, the scores for the various programs are usually smaller than the PURE score but larger than the CHANCE score.

The average scores for the six different programs are given below. These scores are normalized with respect to the PURE score, that is, they give the average drop in percentage points due to the effects of blanks.

METHOD	SCORE
AVERAGE	-9.70
NORMAL	-10.31
LEE1	-11.53
LEE4	-12.53
ZERO	-14.68
DELETE	-17.14

The AVERAGE program seems to be best and the DELETE program seems to be worst. The statistical significance of the difference between AVERAGE and LEE4 is $P=.01$; that is, the probability of the observed difference being due to chance alone is less than 1%. The significance of the difference between LEE1 and DELETE is $P=.1$. The other differences have even less statistical significance.

These P numbers were computed by WILCOXON'S SIGNED RANKS TEST. [8]. This test involves ranking the differences between methods and then computing chance probabilities based on the rank numbers. Since the raw scores are not used it does not matter what sort of statistical distribution the raw scores have. Thus WILCOXON'S SIGNED RANKS TEST is a distribution-free test of statistical significance.

Many commonly used tests of statistical significance assume a normal distribution. When physical measurements are involved it is often reasonable to assume a normal distribution, but in computer work highly exotic distributions are often found therefore it is important to use a distribution-free test.

It is difficult to compare the DELETE method with the others because only the DELETE method discards data. The table below gives the size of the voice data set after the DELETE program has deleted all blanks.

PERCENT BLANKS	M	N
0%	100	11
20%	29	7
30%	19	6
30%	12	7
40%	13	5

It might be said that our scoring method is excessively kind to the DELETE program. Perhaps the deleted vectors should properly be counted "wrong" rather than not counting them at all. If this were done, the scores of the DELETE method would be abysmally low compared to the others. The "correct" way of scoring deleted

vectors depends on the economics of a particular application. One must consider the cost of permitting the luxury of a "don't know" class.

In order to obtain better statistical significance, it was decided to make large scale tests. The results are shown in Table II. Since the DELETE program was already known to be worse than the others, it was not included. Three of the smaller data sets, IRIS, VOICE, and WPS were selected for large scale tests, to conserve computer time. One hundred trials of each data set were made with each of the 5 programs - 1500 jackknife tests altogether. The P figures given in Table II were computed by WILCOXON'S SIGNED RANKS TEST [8] as before.

Table II shows that the ranking of methods depends on the data set. For example, on the IRIS data LEE⁴ is significantly better ($P = .0008$) than LEE¹. While on WPS data the reverse is true ($P = .00001$)

The ZERO method is consistently poor, but the other methods are generally good. When all 300 trials are combined we find that NORMAL appears to be best and the others, in order, are AVERAGE, LEE¹, LEE⁴, ZERO. However, this final ranking must be interpreted cautiously because some of the distinctions lack statistical significance, and also because the relative effectiveness of the various methods depends on the data set.

VII. CONCLUSIONS

1. It is possible to obtain good performance of a pattern recognition system even when the data contains a large percentage of blanks. With 30% blanks inserted into the data, recognition scores will typically drop by 5% to 15%. Exact results depend on the characteristics of the data and on the method used.

2. The DELETE method is very poor and probably should not be used except in an application where there are very few blanks and convenience of implementation outweighs the loss of information.

3. The ZERO method is consistently poor and probably should not be used.

4. NORMAL, AVERAGE, LEE¹, and LEE⁴ are all good methods and one of these might be chosen for a particular application, the choice depending on characteristics of the data, ease of implementation, and other factors.

5. NORMAL and AVERAGE seem to be a little more consistent in performance than LEE¹ and LEE⁴.

6. Because of its consistent performance, ease of implementation and fast running speed we would select NORMAL as the best over all method. However, the other methods might be better in particular applications.

REFERENCES

1. Andrews, Frank M., et. al. "Multiple classification analysis," the University of Michigan, Ann Arbor, 1973, second edition.
2. Cover, T. M., and Hart, P.: "The nearest neighbor decision rule," IEEE Transactions on Information Theory, Vol IT-13, pp. 13-27, Jan, 1967.
3. Duda, R. and Hart, P. (1973) "Pattern classification and scene analysis," Wiley Interscience, N.Y., 1973.
4. Dudani, Sahibsingh A.: "The distance-weighted k-nearest neighbor rule." IEEE Transactions on Systems, Man, and Cybernetics, pp. 325-327, April 1976.
5. Fisher, R. A. "The use of multiple measurements in taxonomic problems," Ann. Eugenics, Part II, pp. 179-188, 1936.
6. Hammer, Michael: "Error detection in data base systems," Proceedings of the National Computer Conference, pp. 795-801, 1976.
7. Hartigan, J. "Clustering Analysis," McGraw-Hill, N.Y. 1975.
8. Langley, Russell, "Practical statistics," Dover, New York, 1970.
9. Lee, R. C. T., Slagle, J. R., Mong., C. T. "Application of clustering to estimate missing data and improve data integrity," Proceedings 2nd International Software Engineering Conference, San Francisco, pp. 539-544, Oct 1976.
10. Meisel, W. S., "Computer-oriented approaches to pattern recognition," Academic Press, N.Y., 1972.
11. Sebestyen, George S. "Decision-making processes in pattern recognition," McMillan, New York, p. 74, 1962.
12. Skinner, W. C.: "A heuristic approach to inductive inference in fact retrieval systems," CACM, Vol. 17, No. 12, pp. 707-711, Dec. 1974.
13. Slagle, J. R., Chang, C. L. and Lee, R. C. T.,: "Experiments with some cluster analysis algorithms," Pattern Recognition, Vol. 6, pp. 181-187, 1974.
14. Slagle, J. R. Chang, C. L. and Heller, S.: "A clustering and data-reorganization algorithm," IEEE Transactions on Systems, Man and Cybernetics, pp. 121-128, Jan 1975.

15. White, E. M. and Fong, P. J., "K-nearest neighbor decision rule performance in speech recognition system," IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-5, p-389, May 1975.

TABLE 1 SCORES OF VARIOUS PROGRAMS

	PURE	CHANCE	LEE4	LEE1	DELETE	NORMAL	AVERAGE	ZERO	DATA SET
1.	95.0	32.2	91.7	86.7	87.5	93.3	93.3	88.3	iris,20%
2.	95.0	32.2	90.0	83.3	87.5	86.7	90.0	75.0	iris,30%
3.	95.0	32.2	83.3	80.0	62.5	83.3	86.7	58.3	iris,40%
4.	95.0	32.2	43.3	56.7	100.0	63.3	58.3	33.3	iris,70%
5.	86.3	13.3	85.0	86.0	85.1	86.3	85.7	86.3	b22,2.8%
6.	86.3	13.3	86.6	84.2	72.4	82.4	81.2	83.0	b22,20%
7.	86.3	13.3	84.2	83.0	31.6	79.1	79.4	79.4	b22,30%
8.	86.3	13.3	80.9	77.9	58.3	78.2	77.6	77.3	b22,40%
9.	64.0	10.5	54.0	62.0	44.8	52.0	57.0	55.0	voice,20%
10.	64.0	10.5	48.0	42.0	36.9	45.0	52.0	43.0	voice,30%
11.	64.0	10.5	50.0	42.0	8.3	45.0	53.0	44.0	voice,30%
12.	64.0	10.5	39.0	32.0	23.1	53.0	50.0	54.0	voice,40%
13.	62.0	33.4	50.0	45.3	58.3	44.7	48.7	48.7	car,20%
14.	62.0	33.4	51.3	59.3	56.9	54.0	57.3	52.0	car,20%
15.	62.0	33.4	46.7	52.0	37.5	48.0	53.3	50.7	car,30%
16.	62.0	33.4	38.7	42.7	66.7	45.3	42.7	47.3	car,40%
17.	59.3	27.7	42.8	48.3	55.3	40.0	48.3	39.3	wps,20%
18.	59.3	27.7	54.5	57.9	43.5	62.1	57.9	59.3	wps,20%
19.	59.3	27.7	45.5	59.3	52.5	61.4	48.3	58.6	wps,30%
20.	59.3	27.7	50.3	55.2	55.0	57.2	51.7	40.0	wps,40%

TABLE II LARGE SCALE TESTS

IRIS DATA		
100 Trials - 30% Blanks		
METHOD	SCORE	P
LEE4	-5.80	.40796
AVERAGE	-6.02	.38139
NORMAL	-6.07	.00147
LEE1	-8.37	.00001
ZERO	-22.30	-----

WPS DATA		
100 Trials - 30% Blanks		
METHOD	SCORE	P
NORMAL	-3.47	.65238
LEE1	-4.36	.00017
AVERAGE	-6.38	.10277
ZERO	-7.73	.01368
LEE4	-11.12	-----

VOICE DATA		
100 trials - 30% blanks		
METHOD	SCORE	P
AVERAGE	-15.54	.94395
LEE4	-15.87	.01040
NORMAL	-18.19	.00113
LEE1	-19.88	.00918
ZERO	-21.33	-----

IRIS + VOICE + WPS		
300 trials - 30% blanks		
METHODS	SCORE	P
NORMAL	-9.24	.27607
AVERAGE	-9.31	.34628
LEE1	-10.87	.00006
LEE4	-10.93	.00895
ZERO	-17.45	-----